Problem Specification

Problem

From part 1:

> The goal of SLAM algorithms (Simultaneous Localization and Mapping) is to map an environment navigated by an autonomous vehicle, while simultaneously locating it in the map, without access to pre-existing maps or external devices. This project will focus on applying SLAM algorithms in the context of sub-aquatic navigation using a sonar device.
>
> The group will have access to sonar data measured by CRAS.

Goals

In this second part of the project, the group intends to expand the existing implementation to 3D space, while focusing on improving performance by employing new algorithms and data structures.

Functionalities

These are the functionalities planned for the second part of the project.

- Map the environment in 3D space:
  - Turn each sonar beam into multiple ray-casts to better simulate the beam's expansion cone.
- Support **incremental map growth**;

Data Structures

The group wants to implement a **hash set** data structure better optimized for the problem at hand;

- Supported operations:
  - **Add** - adds an element to the set;
  - **Remove** - removes an element from the set;
  - **Merge** - merges 2 sets into one;
  - **Difference** - computes the group of elements present in one set, but not in another.

Planned algorithms

- Set collision resolution techniques (focus on *closed hashing (open addressing)*) - compare them:
  - Linear probing;
  - Quadratic probing;
  - Double hashing.
- Explore the ray casting solutions for 3D;
- Explore other image *edge detection* algorithms.

Planned robustness and scalability requirements

Requirements remain similar to the first part of the project. These are:

- **Spacial efficiency** - The sub-aquatic environment can be enormous, as such, the system must store the map in a memory efficient format:
  - In 3D we can better compare the effect of using an Octomap vs a simple *occupancy* grid.
- **Time efficiency** - The system must be efficient in a way that can keep with sonar data in real time:
  - Implementing a better set (comparing to C++ stl implementation), will (hopefully) allow for better times when calculating raycasts for point-clouds.
- **Lightweight** - The system must be robust to limited hardware resources.

Planning

- Problem definition and data structures - Henrique Ribeiro
- Key algorithms - Tiago Silva
- Empirical analysis - João Costa, João Martins